



SGL Reference

Structure Reference

This section introduces structures, variable types and macros that are unique to the SGL. These are essential to programming with the SGL, and the defined contents of each are also important.

Study this reference in addition to the Function Reference.

ATTR

Face attribute list

Structure:

```
typedef struct {  
    Uint8 flag ;  
    Uint8 sort ;  
    Uint16 texno ;  
    Uint16 atrb ;  
    Uint16 colno ;  
    Uint16 gstb ;  
    Uint16 dir ;  
} ATTR ;
```

Members:

flag	Front/back setting
sort	Sort setting
texno	Texture number
atr	Attribute data
colno	Color number
gstb	Gouraud setting
dir	Texture reversal setting and type

Description:

This structure defines the polygon face attribute list.

Remarks:

for details on the face attributes and how to use them, refer to chapter 7, "Polygon Face Attributes," in the Programmer's Tutorials.

EVENT

Event management

Structure:

```
typedef struct evnt {
    WORK *work ;
    struct event *next ;
    struct event *before ;
    void (*exad)() ;
    Uint8 user[EVENT_SIZE-(sizeof(WORK*)
        +sizeof(struct evnt)*2+sizeof(void (*)()))]
}EVENT;
```

Members:

work Work area pointer
next Starting address of next event
before Starting address of previous event
(exad)() Function execution address
user[] Work area

Description:

This structure defines the event management table. Set the address received from the library function "slGetWork" in the member "work".

The default value is NULL.

Remarks

"EVENT_SIZE" in the member "user[]" is 128 bytes. As a result, the user area is 112 bytes.

PDATA

Polygon model data

Structure

```
typedef struct {  
    POINT *pntb1 ;  
    Uint32 nbPoint ;  
    POLYGON *pltb1 ;  
    Uint32 nbPolygon ;  
    ATTR *attb1 ;  
} PDATA ;
```

Members

pntb1 Vertex list pointer
nbPointNumber of vertices
pltb1 Face list pointer
nbPolygon Number of faces
attb1 Attribute list pointer

Description:

This structure defines the polygon model data.

Remarks

PICTURE

Texture registration table

Structure	<pre>typedef struct { Unit16 texno ; Unit16 cmode ; void *pcsrc ; } PICTURE ;</pre>
Members	<p>texno Texture number</p> <p>cmode Color mode</p> <p>pcsrc Starting address of texture data to be registered</p>
Description	<p>This structure is the information table for transferring and registering texture data within VRAM.</p>
Remarks	<p>Refer to the textures in the "Polygon Face Attributes."</p>

ROTSCROLL

Rotation parameters

STRUCTURE

```
typedef struct {
    FIXED XST ;
    FIXED YST ;
    FIXED ZST ;
    FIXED DXST ;
    FIXED DYST ;
    FIXED DX ;
    FIXED DY ;
    FIXED MATA ;
    FIXED MATB ;
    FIXED MATC ;
    FIXED MATD ;
    FIXED MATE ;
    FIXED MATF ;
    Sint16 PX ;
    Sint16 PY ;
    Sint16 pZ ;
    Sint16 dummy0 ;
    Sint16 CX ;
    Sint16 CY ;
    Sint16 CZ ;
    Sint 16 dummy1 ;
    FIXED MX ;
    FIXED MY ;
    FIXED KX ;
    FIXED KY ;
    Uint32 KAST ;
    Sint32 DKA ;
}ROTSCROLL ;
```

Members

XST	Scroll screen start coordinate Xst
YST	Scroll screen start coordinate Yst
ZST	Scroll screen start coordinate Zst
DXST	Scroll screen vertical direction coordinate increment amount dXst
DYST	Scroll screen vertical direction coordinate increment amount dYst
DX	Scroll screen horizontal direction coordinate increment amount dX
DY	Scroll screen horizontal direction coordinate increment amount dY
MATA	Rotating matrix parameter A
MATB	Rotating matrix parameter B
MATC	Rotating matrix parameter C
MATD	Rotating matrix parameter D
MATE	Rotating matrix parameter E
MATF	Rotating matrix parameter F
PX	Viewpoint coordinate Px
PY	Viewpoint coordinate Py
PZ	Viewpoint coordinate Pz
dummy0	Dummy area
CX	Center coordinate Px
CY	Center coordinate Py

CZ Center coordinate Pz
dummy1 Dummy area
MX Parallel movement amount Mx
MY Parallel movement amount My
KX Enlargement/reduction coefficient kx
KY Enlargement/reduction coefficient ky
KAST Coefficient table start address KAsT
DKAST Coefficient table vertical direction address increment DKAsT
DKA Coefficient table vertical direction address increment DKA

Description

This structure defines the rotation parameter table.

The rotation parameter table is read for each line of the rotating scroll screen, and the screen is displayed according to those values.

Remarks

Used together with the NORMAL macro, this macro is used to make the POLYGON face list.

SPR_ATTR

Sprite attributes table

STRUCTURE

```
typedef struct spratr {
    Uint16 texno ;
    Uint16 atrb ;
    Uint16 colno ;
    Uint16 gstb ;
    Uint16 dir ;
} SPR_ATTR ;
```

Members

texno	Texture number
atrb	Attribute data (display mode)
colno	Color number
gstb	Gouraud shading table
dir	Texture reversal

Description

This structure is the parameter table for sprite display. Basically, these parameters conform with the texture parameters.

Remarks

Refer to the group of functions concerning sprite display.

SPRITE

Sprite data

STRUCTURE

```
typedef struct {
    Uint16 CTRL ;
    Uint16 LINK ;
    Uint16 PMOD ;
    Uint16 COLR ;
    Uint16 SRCA ;
    Uint16 SIZE ;
    Uint16 XA ;
    Uint16 YA ;
    Uint16 XB ;
    Uint16 YB ;
    Uint16 XC ;
    Uint16 YC ;
    Uint16 XD ;
    Uint16 YD ;
    Uint16 GRDA ;
    Uint16 DMMY ;
} SPRITE ;
```

Members

CTRL Control function
 LINK Link address
 PMOD Put mode
 COLR Color data
 SRCA CG address
 SIZE Character size
 XA X coordinate of display position A
 YA Y coordinate of display position A
 XB X coordinate of display position B
 YB Y coordinate of display position B
 XC X coordinate of display position C
 YC Y coordinate of display position C
 XD X coordinate of display position D
 YD Y coordinate of display position D
 GRDA Gouraud shading table
 DMMY Dummy data used to match up with size

Description

This data type is used to directly pass data to the VDP1, and is the data table used to display sprites.

Sprite picture data must be stored and registered in VRAM beforehand. (The same is true for texture data and scroll data.) For details, refer to the textures in "Polygon Face Attributes".

Remarks

Textures and sprites used in the Sega Saturn system have very similar data structures, but their display methods differ as follows:

Textures are applied to polygon faces and displayed.

Sprites are displayed independently.

TEXTURE

Texture data

STRUCTURE

```
typedef struct {
    Uint16 Hsize ;
    Uint16 Vsize ;
    Uint16 CGadr ;
    Uint16 HVsize ;
} TEXTURE ;
```

Members

Hsize Horizontal size of texture
 Vsize Vertical size of texture
 CGadr CG address of texture/8
 HVsize Horizontal size/8, vertical size (for hardware) ((HSIZE/8)<<8)|(V SIZE)

Description

This structure is the texture management table that is needed in order to use textures in the SGL.

Remarks

For details, refer to chapter 7, "Polygon Face Attributes", in the Programmer's Tutorial.

WORK

Work area management

STRUCTURE

```
typedef struct work {
    struct work *next ;
    Uint8 user[WORK_SIZE-sizeof(struct work *)] ;
} WORK ;
```

Members

next Pointer to next work area
user[] Free area within the work area that the user can use

Description

This structure indicates a work area that can be used within an event. The member "user" consists of the work size (WORK_SIZE = 64 bytes) less the size of the member "next" (4 bytes), for a total of 60 bytes.

For details, refer to the Chapter 10, "Event Control" in the Programmer's Tutorial.

Remarks

Refer to: Chapter 10, "Event Control"

WORK

SmpcDateTime

RTC time

STRUCTURE

```
typedef struct {
    Uint16 year ;
    Uint8  month ;
    Uint8  date  ;
    Uint8  hour  ;
    Uint8  minute ;
    Uint8  second ;
} SmpcDateTime ;
```

Members

year	Year
month	Day of the week and month
date	Date
hour	Hours
minute	Minutes
second	Seconds

Description

This structure is used to reference the RTC time.

Remarks

Use this structure when referencing the "rtc" member of the system variable "Smpc_Status".

SmpcStatus

SMPC status

STRUCTURE

```
typedef struct {
    Uint8      cond ;
    SmpcDateTime month ;
    Uint8      ctg :
    Uint8      area ;
    Uint16     system ;
    Uint32     smem ;
} SmpcStatus ;
```

Members

```
cond   Status
rtc     RTC time
ctg     Cartridge code
area    Area code
system  System status
smem    SMPC memory data
```

Description

This structure is used to reference the SMPC system status.

Remarks

Use this structure when referencing the system variable "Smpc_Status".

The member "rtc" can be referenced as the structure "SmpcDateTime".

Special get and set functions are provided for the member "smem".

PerDigital

Digital device

STRUCTURE

```
typedef struct {  
    Uint8    id ;  
    Uint8    ext ;  
    Uint16   data ;  
    Uint16   push ;  
    Uint16   pull ;  
} PerDigital ;
```

Members

id	Peripheral ID
ext	Extended data size
data	Current button data
push	Depressed button data
pull	Unpressed button data

Description

This structure is used to reference digital devices.

Remarks

Use this structure when referencing the system variable "Smpc_Peripheral". All devices can be handled as digital devices.

PerAnalog

Analog device

STRUCTURE

```
typedef struct {
    Uint8    id ;
    Uint8    ext ;
    Uint16   data ;
    Uint16   push ;
    Uint16   pull ;
    Uint8    x ;
    Uint8    y ;
    Uint8    z ;
} PerAnalog ;
```

Members

id	Peripheral ID
ext	Extended data size
data	Current button data
push	Depressed button data
pull	Unpressed button data
x	Absolute value of X axis data
y	Absolute value of Y axis data
z	Absolute value of Z axis data

Description

This structure is used to reference analog devices.

Remarks

Type-cast the system variable "Smpc_Peripheral" and use this structure to reference a peripheral as an analog device.

PerPoint

Pointing device

STRUCTURE

```
typedef struct {
    Uint8    id ;
    Uint8    ext ;
    Uint16   data ;
    Uint16   push ;
    Uint16   pull ;
    Uint16   x ;
    Uint16   y ;
} PerPoint ;
```

Members

id	Peripheral ID
ext	Extended data size
data	Current button data
push	Depressed button data
pull	Unpressed button data
x	X coordinate
y	Y coordinate

Description

This structure is used to reference a pointing device.

Remarks

Type-cast the system variable "Smpc_Peripheral" and use this structure to reference a peripheral as a pointing device.

PerKeyBoard

Keyboard device

STRUCTURE

```
typedef struct {
    Uint8    id ;
    Uint8    ext ;
    Uint16   data ;
    Uint16   push ;
    Uint16   pull ;
    Uint8    cond ;
    Uint8    code ;
} PerKeyBoard ;
```

Members

id	Peripheral ID
ext	Extended data size
data	Current button data
push	Depressed button data
pull	Unpressed button data
cond	Status data
code	Key code

Description

This structure is used to reference the keyboard device.

Remarks

Type-cast the system variable "Smpc_Peripheral" and use this structure to reference a peripheral as a keyboard device.

ANGLE

Angle data variable type

Structure

```
typedef Sint16 ANGLE ;
```

Members

Description

Angle data notation variable type.
The range from 0_i to 359_i is expressed by 0x0000 to 0xffff.

Remarks

FIXED

Fixed-point decimal variable type

Structure

```
typedef Sint32 FIXED ;
```

Members

Description

This variable type indicates fixed-point decimal data. FIXED-type values are represented in the following manner.

High-order 16 bits: Integer portion

Low-order 16 bits: Decimal portion

Ex.: 16.5 -> 0x00108000

Remarks

MATRIX

Matrix variable type

Structure	<pre>typedef FIXED MATRIX[4][3] ;</pre>
Members	
Description	Matrix notation variable type
Remarks	<p>FIXED matrix [4][3]; MATRIX matrix; The two definitions shown above have the same meaning.</p>

POINT

Vertex data type

Structure

```
typedef FIXED POINT[xyz] ;
```

Members

Description

This variable type defines the vertex data used in polygons.

Remarks

FIXED point [xyz];
FIXED point [3];
POINT point;
All three of the above definitions have the same meaning.

TEXDAT

Texture data variable type

Structure	<pre>typedef UINT16 TEXDAT ;</pre>
Members	
Description	<p>This variable type is used to define the actual texture itself.</p>
Remarks	

VECTOR

Vector variable type

Structure

TYPED FIXED VECTOR[XYZ]

Members

Description

Vector data variable type

Remarks

FIXED vector[XYZ];
FIXED vector[3];
VECTOR vector
The above three definitions all have the same meaning.

ATTRIBUTE

Polygon attribute specification

Structure

```
#define ATTRIBUTE(plane,sort,texture,color,gourand,mode,dir,option)
    _@_@_@_@_@ {
        plane,(sort)|(((dir)>>16)&0x01c)|(option),__
        texture,(mode)|(((dir)>>24)&0xc0,color__
        gourud,(dir)&0x03f
    }
    Uint8 plane ;
    Uint8 sort ;
    Uint16 texture ;
    Uint16 color ;
    Uint16 gouraud ;
    Uint 16 mode ;
    Uint 32 dir ;
    Uint16 option :
```

Members

plane Front/back attribute
 sort Z sort specification
 texture Texture number, or No_Texture
 color C_RGB macro-specified color, color palette number, or No_Palet
 gouraud Gouraud table, or No_Gouraud
 mode Various mode specifications for the polygon
 dir Specification of texture display direction, etc.
 option Other settings for the polygon

Description

This macro sets the face attributes (particularly the polygon front face) concerning polygon drawing. For details on the meaning of and substitution values for each parameter, refer to chapter 7, "Polygon Face Attributes," in the Programmer's Tutorial. Also refer to the list of ATTRIBUTE macro substitution values at the end of the Structure Reference.

Remarks

When using texture, the member color is sometimes used to specify the color bank number.

C_RGB

RGB value specification

Structure

```
#define C_RGB(r,g,b) (((b)&0x1f)<<10|((g)&0x1f)<<5|((r)&0x1f) |0x8000)
    Uint8r ;
    Uint8 g ;
    Uint8 b ;
```

Members

```
r    Red
g    Green
b    Blue
```

Description

This macro specifies the RGB values used to represent color gradations. The color gradation values can range from 0 to 1f for each of red, green, and blue.

Remarks

This macro cannot be used to specify the transparent color.

DEGtoANG

Angle conversion macro

Structure

```
#define DEGtoANG(d) (ANGLE)((d)*65536.0/360.0)
float d ;
```

Members

d Angle to be converted (DEG notation)

Description

This macro converts a floating-point angle value expressed in DEG notation to an ANGLE-type value.

Remarks

NORMAL

Coordinate value conversion macro

Structure	<pre>#define NORMAL(x,y,z) {POStoFIXED(x,y,z)</pre>
Members	<div> <div>x</div> <div>X coordinate to be converted</div> </div> <div> <div>y</div> <div>Y coordinate to be converted</div> </div> <div> <div>z</div> <div>Z coordinate to be converted</div> </div>
Description	<p>This macro converts a normal vector XYZ coordinates expressed by floating-point decimals into FIXED-type variables.</p>
Remarks	<p>Used together with the VERTICES macro, this macro is used to make the POLYGON face list.</p>

Refer to:

NORMAL

PICDEF

Texture management table

Structure	<pre>#define PICDEF(texno,cmode,pcsrc) {(Uint16)(texno),(Uint16)(cmde), (void*)(pcsrc)} Uint16 texno ; Uint16 cmode ; void*pcsrc ;</pre>
Members	<p>texno Texture number</p> <p>cmode Color mode (COL_16, 64, 128, 256, or 32K)</p> <p>pcsrc Pointer for texture data defined by "TEXDAT"</p>
Description	<p>This macro creates the table of information used to set a texture in VRAM so that the texture can be handled within a program.</p>
Remarks	

POStoFIXED

Coordinate value conversion macro

Structure

```
#define POStoFIXED(x,y,z) {toFIXED(x),(toFIXED(y),toFIXED(z)}
```

Members

x	X coordinate to be converted
y	Y coordinate to be converted
z	Z coordinate to be converted

Description

This macro converts the XYZ coordinate values to FIXED-type variables.

Remarks

TEXDEF

Texture registration table

Structure	<pre>#define TEXDEF(h, v, presize) (h, v, (((cgaddress + (presize)) * 4) >> pal) / 8, (((h) & 0x1f8) << 5 (v))) Uint16h ; Uint16v ; Uint32 presize ;</pre>
Members	<p>h Horizontal size of texture</p> <p>v Vertical size of texture</p> <p>presize Previously registered texture size (vertical x horizontal)</p>
Description	<p>This macro creates a table for getting texture information.</p>
Remarks	<p>Reference macros</p> <pre>#define cgaddress 0x10000 #define pal COL_32K</pre>

toFIXED

Value conversion macro

Structure

```
#define toFIXED(a) ((FIXED)((a)*65536.0)
```

Members

a Value to be converted

Description

This macro converts the value supplied as the parameter into a FIXED-type value.

Remarks

VERTICES

Polygon vertex variable string

Structure

```
#define VERTICES(v0, v1, v2, v3) (v0, v1, v2, v3)}  
    Uint16 v0 ;  
    Uint16 v1 ;  
    Uint16 v2 ;  
    Uint16 v3 ;
```

Members

v0	Vertex v0
v1	Vertex v1
v2	Vertex v2
v3	Vertex v3

Description

This macro specifies the polygon vertex numbers expressed as integers.

Remarks

Used together with the NORMAL macro, this macro is used to make the polygon face list.